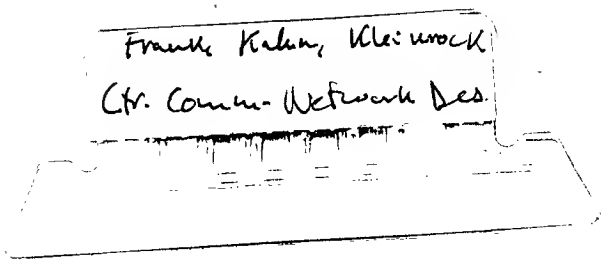
Frank Kahn, Kleinrock
Ctr. Comm. Network Res.

Computer communication network design—Experience with theory and practice*

by HOWARD FRANK
Network Analysis Corporation
Glen Cove, New York

ROBERT E. KAHN
Bolt Beranek and Newman Inc.
Cambridge, Massachusetts

and

LEONARD KLEINROCK
University of California
Los Angeles, California

INTRODUCTION

The ARPA Network (ARPANET) project brought together many individuals with diverse backgrounds, philosophies, and technical approaches from the fields of computer science, communication theory, operations research and others. The project was aimed at providing an efficient and reliable computer communications system (using message switching techniques) in which computer resources such as programs, data, storage, special purpose hardware etc., could be shared among computers and among many users.[38] The variety of design methods, ranging from theoretical modeling to hardware development, were primarily employed independently, although cooperative efforts among designers occurred on occasion. As of November, 1971, the network has been an operational facility for many months, with about 20 participating sites, a network information center accessible via the net, and well over a hundred researchers, system programmers, computer center directors and other technical and administrative personnel involved in its operation.

In this paper, we review and evaluate the methods used in the ARPANET design from the vantage of over two years' experience in the development of the network. In writing this paper, the authors have each made equal contributions during a series of intensive discussions and debates. Rather than present merely a summary of the procedures that were used in the network design, we have attempted to evaluate each other's methods to determine their advantages and drawbacks. Our approaches and philosophies have often differed radically and, as a result, this has not been an easy or undisturbing process. On the other hand, we have found our collaboration to be extremely rewarding and, notably, we have arrived at many similar conclusions about the network's behavior that seem to be generally applicable to message switched networks.

The essence of a network is its design philosophy, its performance characteristics, and its cost of implementation and operation. Unfortunately, there is no generally accepted definition of an "optimal" network or even of a "good" network. For example, a network designed to transmit large amounts of data only during late evening hours might call for structural and performance characteristics far different from one servicing large numbers of users who are rapidly exchanging short messages during business hours. We expect this topic, and others such as the merits of message switching vs. circuit switching or distributed vs. centralized control to be a subject of discussion for many years.[1,14,24,32,34,37]

A cost analysis performed in 1967-1968 for the ARPA Network indicated that the use of message switching would lead to more economical communications and better overall availability and utilization of resources than other methods.[36,38] In addition to its impact on the availability of computer resources, this decision has generated widespread interest in store-and-forward communications. In many instances, the use of store-and-forward communication techniques can result in

* This work was supported by the Advanced Research Projects Agency of the Department of Defense under Contract No. DAHC 15-70-C-0120 at the Network Analysis Corporation, Contract Nos. DAHC 15-69-C-0179 and DAHC-71-C-0088 at Bolt Beranek and Newman Inc., and Contract No. DAHC 15-69-C-0285 at the University of California at Los Angeles.

greater flexibility, higher reliability, significant technical advantage, and substantial economic savings over the use of conventional common carrier offerings. An obvious trend toward increased computer and communication interaction has begun. In addition to the ARPANET, research in several laboratories is under way, small experimental networks are being built, and a few examples of other government and commercial networks are already apparent.[6,7,31,40,41,47,48,52]

In the ARPANET, each time-sharing or batch processing computer, called a Host, is connected to a small computer called an Interface Message Processor (IMP). The IMPs, which are interconnected by leased 50 kilobit/second circuits, handle all network communication for their Hosts. To send a message to another Host, a Host precedes the text of its message with an address and simply delivers it to its IMP. The IMPs then determine the route, provide error control, and notify the sender of its receipt. The collection of Hosts, IMPs, and circuits forms the message switched resource sharing network. A good description of the ARPANET, and some early results on protocol development and modeling are given in References 3, 12, 15, 23 and 38. Some experimental utilization of the ARPANET is described in Reference 42. A more recent evaluation of such networks and a forward look is given in References 35 and 39.

The development of the Network involved four principal activities:

(1) The design of the IMPs to act as nodal store-and-forward switches,
(2) The topological design to specify the capacity and location of each communication circuit within the network,
(3) The design of higher level protocols for the use of the network by time-sharing, batch processing and other data processing systems, and
(4) System modeling and measurement of network performance.

Each of the first three activities were essentially performed independently of each other, whereas the modeling effort partly affected the IMP design effort, and closely interacted with the topological design project.

The IMPs were designed by Bolt Beranek and Newman Inc. (BBN) and were built to operate independent of the exact network connectivity; the topological structure was specified by Network Analysis Corporation (NAC) using models of network performance developed by NAC and by the University of California at Los Angeles (UCLA). The major efforts in the area of system modeling were performed at UCLA using theoretical and simulation techniques. Network performance measurements have been conducted during the development of the network by BBN and by the Network Measurement Center at UCLA. To facilitate effective use of the net, higher level (user) protocols are under development by a group of representatives of universities and research centers. This group, known as the Network Working Group, has already specified a Host to Host protocol and a Telnet protocol, and is in the process of completing other function oriented protocols.[4,29] We make no attempt to elaborate on the Host to Host protocol design problems in this paper.

THE NETWORK DESIGN PROBLEM

A variety of performance requirements and system constraints were considered in the design of the net. Unfortunately, many of the key design objectives had to be specified long before the actual user requirements could be known. Once the decision to employ message switching was made, and fifty kilobit/second circuits were chosen, the critical design variables were the network operating procedure and the network topology; the desired values of throughput, delay, reliability and cost were system performance and constraint variables. Other constraints affected the structure of the network, but not its overall properties, such as those arising from decisions about the length of time a message could remain within the network, the location of IMPs relative to location of Hosts, and the number of Hosts to be handled by a single IMP.

In this section, we identify the central issues related to IMP design, topological design, and network modeling. In the remainder of the paper, we describe the major design techniques which have evolved.

IMP properties

The key issue in the design of the IMPs was the definition of a relationship between the IMP subnet and the Hosts to partition responsibilities so that reliable and efficient operation would be achieved. The decision was made to build an autonomous subnet, independent (as much as possible) of the operation of any Host. The subnet was designed to function as a "communications system"; issues concerning the use of the subnet by the Hosts (such as protocol development) were initially left to the Hosts. For reliability, the IMPs were designed to be robust against all line failures and the vast majority of IMP and Host failures. This decision required routing strategies that dynamically adapt to changes in the states of IMPs and circuits,

and an elaborate flow control strategy to protect the subnet against Host malfunction and congestion due to IMP buffer limitations. In addition, a statistics and status reporting mechanism was needed to monitor the behavior of the network.

The number of circuits that an IMP must handle is a design constraint directly affecting both the structure of the IMP and the topological design. The speed of the IMP and the required storage for program and buffers depend directly upon the total required processing capacity, which must be high enough to switch traffic from one line to another when all are fully occupied. Of great importance is the property that all IMPs operate with identical programs. This technique greatly simplifies the problem of network planning and maintenance and makes network modifications easy to perform.

The detailed physical structure of the IMP is not discussed in this paper.[2,15] However, the operating procedure, which guides packets through the net, is very much of interest here. The flow control, routing, and error control techniques are integral parts of the operating procedure and can be studied apart from the hardware by which they are implemented. Most hardware modifications require changes to many IMPs already installed in the field, while a change in the operating procedure can often be made more conveniently by a change to the single operating program common to all IMPs, which can then be propagated from a single location via the net.

Topological properties

The topological design resulted in the specification of the location and capacity of all circuits in the network. Projected Host—Host traffic estimates were known at the start to be either unreliable or wrong. Therefore, the network was designed under the assumption of equal traffic between all pairs of nodes. (Additional superimposed traffic was sometimes included for those nodes with expectation of higher traffic requirements.) The topological structure was determined with the aid of specially developed heuristic programs to achieve a low cost, reliable network with a high throughput and a general insensitivity to the exact traffic distribution. Currently, only 50 kilobit/second circuits are being used in the ARPANET. This speed line was chosen to allow rapid transmission of short messages for interactive processing (e.g., less than 0.2 seconds average packet delay) as well as to achieve high throughput (e.g., at least 50 kilobits/second) for transmission of long messages. For reliability, the network was constrained to have at least two independent paths between each pair of IMPs.

The topological design problem requires consideration of the following two questions:

(1) Starting with a given state of the network topology, what circuit modifications are required to add or delete a set of IMPs?
(2) Starting with a given state of network topology, when and where should circuits be added or deleted to account for long term changes in network traffic?

If the locations of all network nodes are known in advance, it is clearly most efficient to design the topological structure as a single global effort. However, in the ARPANET, as in most actual networks, the initial designation of node locations is modified on numerous occasions. On each such occasion, the topology can be completely reoptimized to determine a new set of circuit locations.

In practice, there is a long lead time between the ordering and the delivery of a circuit, and major topological modifications cannot be made without substantial difficulty. It is therefore prudent to add or delete nodes with as little disturbance as possible to the basic network structure consistent with overall economical operation. Figure 1 shows the evolution of the ARPANET from the basic four IMP design in 1969 to the presently planned 27 IMP version. Inspection of the 24 and 27 IMP network designs reveals a few substantial changes in topology that take advantage of the new nodes being added. Surprisingly enough, a complete "reoptimization" of the 27 IMP topology yields a network only slightly less expensive (about 1 percent) than the present network design.[28]

Network models

The development of an accurate mathematical model for the evaluation of time delay in computer networks is among the more difficult of the topics discussed in this paper. On the one hand, the model must properly reflect the relevant features of the network structure and operation, including practical constraints. On the other hand, the model must result in a mathematical formulation which is tractable and from which meaningful results can be extracted. However, the two requirements are often incompatible and we search for an acceptable compromise between these two extremes.

The major modeling effort thus far has been the study of the behavior of networks of queues.[21] This emphasis is logical since in message switched systems, messages experience queueing delays as they pass from node to node and thus a significant performance measure is the

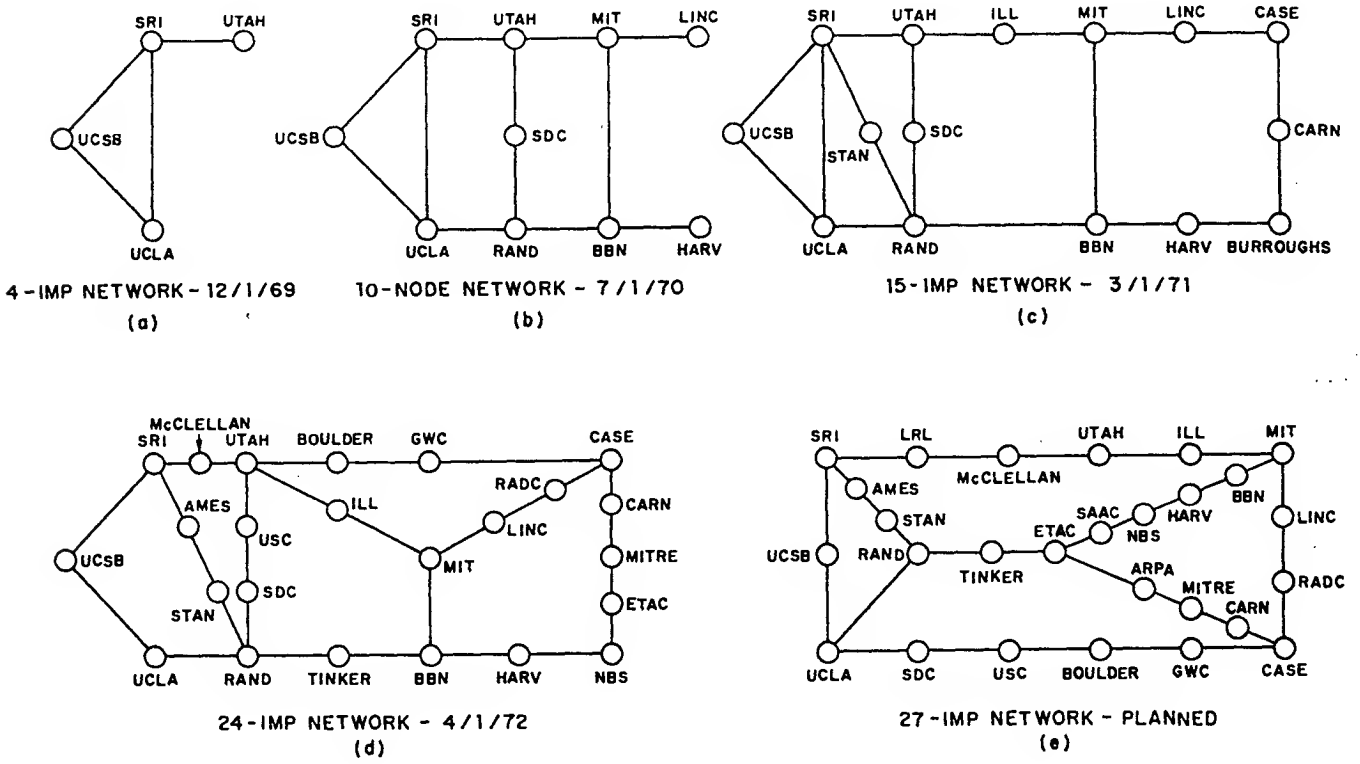

Figure 1—The evolution of the ARPANET

speed at which messages can be delivered. The queueing models were developed at a time when there were no operational networks available for experimentation and model validation, and simulation was the only tool capable of testing their validity. The models, which at all times were recognized to be idealized statements about the real network, were nonetheless crucial to the ARPANET topological design effort since they afforded the only known way to quantitatively predict the properties of different routing schemes and topological structures. The models have been subsequently demonstrated to be very accurate predictors of network throughput and indispensable in providing analytical insight into the network's behavior.

The key to the successful development of tractable models has been to factor the problem into a set of simpler queueing problems. There are also heuristic design procedures that one can use in this case. These procedures seem to work quite well and are described later in the paper. However, if one specializes the problem and removes some of the real constraints, theory and analysis become useful to provide understanding, intuition and design guidelines for the original constrained problem. This approach uncovers global properties of network behavior, which provide keys to good heuristic design procedures and ideal performance bounds.

DESIGN TECHNIQUES

In this section we describe the approaches taken to the design problems introduced in the previous section. We first summarize the important properties of the ARPANET design:

(1) A communications cost of less than 30 cents per thousand packets (approximately a megabit).
(2) Average packet delays under 0.2 seconds through the net.
(3) Capacity for expansion to 64 IMPs without major hardware or software redesign.
(4) Average total throughput capability of 10-15 kilobits/second for all Hosts at an IMP.
(5) Peak throughput capability of 85 kilobits/second per pair of IMPs in an otherwise unloaded network.
(6) Transparent communications with maximum message size of approximately 8000 bits and error rates of one bit in 10^{12} or less.

(7) Approximately 98 percent availability of any IMP and close to 100 percent availability of all operating IMPs from any operable IMP.

The relationships between the various design efforts are illustrated by these properties. The topological design provides for both a desired average throughput and for two or more paths to be fully used for communication between any pair of Hosts. The operating procedure should allow any pair of Hosts to achieve those objectives. The availability of IMPs to communicate reflects both the fact that IMPs are down about 2 percent of the time and that the topology is selected so that circuit failures contribute little additional to the total system downtime.

IMP design

The IMP design consists of two closely coupled but nonetheless separable pieces—the physical hardware specification (based on timing and reliability considerations and the operating procedure) and the design and implementation of the operating procedure using the specified IMP hardware. The IMP originally developed for the ARPANET contains a 16-bit one microsecond computer that can handle a total of about ¾ megabits/second of "useful" information on a total of approximately one megabit/second of circuit capacity (e.g., twenty 50 kilobit/second circuits). Hardware is likely to change as a function of the required IMP capacity but an operating procedure that operates well at one IMP capacity is likely to be transferable to machines that provide different capacity. However, as a network grows in size and utilization, a more comprehensive operating procedure that takes account of known structural properties, such as a hierarchical topology, is appropriate.

Four primary areas of IMP design, namely message handling and buffering, error control, flow control, and routing are discussed in this section. The IMP provides buffering to handle messages for its Host and packets for other IMPs. Error control is required to provide reliable communication of Host messages in the presence of noisy communication circuits. The design of the operating procedure should allow high throughput in the net under heavy traffic loads. Two potential obstacles to achieving this objective are: (1) The net can become congested and cause the throughput to *decrease* with increasing load, and (2) The routing procedure may be unable to always adapt sufficiently fast to the rapid movement of packets to insure efficient routing. A flow control and routing procedure is needed that can efficiently meet this requirement.

Message handling and buffering

In the ARPANET, the maximum message size was constrained to be approximately 8000 bits. A pair of Hosts will typically communicate over the net via a sequence of transmitted messages. To obtain delays of a few tenths of a second for such messages and to lower the required IMP buffer storage, the IMP program partitions each message into one or more packets each containing at most approximately 1000 bits. Each packet of a message is transmitted independently to the destination where the message is reassembled by the IMP before shipment to that destination Host. Alternately, the Hosts could assume the responsibility for reassembling messages. For an asynchronous IMP-Host channel, this marginally simplifies the IMP's task. However, if *every* IMP-Host channel were synchronous, and the Host provided the reassembly, the IMP task can be further simplified. In this latter case, "IMP-like" software would have to be provided in each Host.

The method of handling buffers should be simple to allow for fast processing and a small amount of program. The number of buffers should be sufficient to store enough packets for the circuits to be used to capacity; the size of the buffers may be intuitively selected with the aid of simple analytical techniques. For example, fixed buffer sizes are useful in the IMP for simplicity of design and speed of operation, but inefficient utilization can arise because of variable length packets. If each buffer contains A words of overhead and provides space for M words of text, and if message sizes are uniformly distributed between 1 and L, it can be shown[45] that the choice of M that minimizes the expected storage is approximately $\sqrt{AL}$. In practice, M is chosen to be somewhat smaller on the assumption that most traffic will be short and that the amount of overhead can be as much as, say, 25 percent of buffer storage.

Error control

The IMPs must assume the responsibility for providing error control. There are four possibilities to consider:

(1) Messages are delivered to their destination out of order.

(2) Duplicate messages are delivered to the destination.

(3) Messages are delivered with errors.

(4) Messages are not delivered.

The task of proper sequencing of messages for delivery to the destination Host actually falls in the province of both error control and flow control. If at most one message at a time is allowed in the net between a pair of Hosts, proper sequencing occurs naturally. A duplicate packet will arrive at the destination IMP after an acknowledgment has been missed, thus causing a successfully received packet to be retransmitted. The IMPs can handle the first two conditions by assigning a sequence number to each packet as it enters the network and processing the sequence number at the destination IMP. A Host that performs reassembly can also assign and process sequence numbers and check for duplicate packets. For many applications, the order of delivery to the destination is immaterial. For priority messages, however, it is typically the case that fast delivery requires a perturbation to the sequence.

Errors are primarily caused by noise on the communication circuits and are handled most simply by error detection and retransmission between each pair of IMPs along the transmission path. This technique requires extra storage in the IMP if either circuit speeds or circuit lengths substantially increase. Failures in detecting errors can be made to occur on the order of years to centuries apart with little extra overhead (20-30 parity bits per packet with the 50 kilobit/second circuits in the ARPANET). Standard cyclic error detection codes have been usefully applied here.

A reliable system design insures that each transmitted message is accurately delivered to its intended destination. The occasional time when an IMP fails and destroys a useful in-transit message is likely to occur far less often than a similar failure in the Hosts and has proven to be unimportant in practice, as are errors due to IMP memory failures. A simple end to end retransmission strategy will protect against these situations, if the practical need should arise. However, the IMPs are designed so that they can be removed from the network without destroying their internally stored packets.

Flow control

A network in which packets may freely enter and leave can become congested or logically deadlocked and cause the movement of traffic to halt.[5,17] Flow control techniques are required to prevent these conditions from occurring. The provision of extra buffer storage will mitigate against congestion and deadlocks, but cannot in general prevent them.

The sustained failure of a destination Host to accept packets from its IMP at the rate of arrival will cause the net to fill up and become congested. Two kinds of logical deadlocks, known as reassembly lockup and store-and-forward lockup may also occur. In reassembly lockup, the remaining packets of partially reassembled messages are blocked from reaching the destination IMP (thus preventing the message from being completed and the reassembly space freed) by other packets in the net that are waiting for reassembly space at that destination to become free. In a store-and-forward lockup, the destination has room to accept arriving packets, but the packets interfere with each other by tying up buffers in transit in such a way that none of the packets are able to reach the destination.[17] These phenomena have only been made to occur during very carefully arranged testing of the ARPANET and by simulation.[49]

In the original ARPANET design, the use of software links and RFNMS protected against congestion by a single link or a small set of links. However, the combined traffic on a large number of links could still produce congestion. Although this strategy did not protect against lockup, the method has provided ample protection for the levels of traffic encountered by the net to date.

A particularly simple flow control algorithm that augments the original IMP design to prevent congestion and lockup is also described in Reference 17. This scheme includes a mechanism whereby packets may be discarded from the net at the destination IMP when congestion is about to occur, with a copy of each discarded packet to be retransmitted a short time later by the originating Host's IMP. Rather than experience excessive delays within the net as traffic levels are increased, the traffic is queued outside the net so that the transit time delays internal to the net continue to remain small. This strategy prevents the insertion of more traffic into the net than it can handle.

It is important to note the dual requirement for small delays for interactive traffic and high bandwidth for the fast transfer of files. To allow high bandwidth between a pair of Hosts, the net must be able to accept a steady flow of packets from one Host and at the same time be able to rapidly quench the flow at the entrance to the source IMP in the event of imminent congestion at the destination. This usually requires that a separate provision be made in the algorithm to protect short interactive messages from experiencing unnecessarily high delays.

Routing

Network routing strategies for distributed networks require routing decisions to be made with only information available to an IMP and the IMP must

execute those decisions to effect the routing.[14,15] A simple example of such a strategy is to have each IMP handling a packet independently route it along its current estimate of the shortest path to the destination.

For many applications, it suffices to deal with an idealized routing strategy which may not simulate the IMP routing functions in detail or which uses information not available to the IMP. The general properties of both strategies are usually similar, differing mainly in certain implementation details such as the availability of buffers or the constraint of counters and the need for the routing to quickly adapt to changes in IMP and circuit status.

The IMPs perform the routing computations using information received from other IMPs and local information such as the alive/dead state of its circuits. In the normal case of time varying loads, local information alone, such as the length of internal queues, is insufficient to provide an efficient routing strategy without assistance from the neighboring IMPs. It is possible to obtain sufficient information from the neighbors to provide efficient routing, with a small amount of computation needed per IMP and without each IMP requiring a topological map of the network. In certain applications where traffic patterns exhibit regularity, the use of a central controller might be preferable. However, for most applications which involve dynamically varying traffic flow, it appears that a central controller cannot be used more effectively than the IMPs to update routing tables if such a controller is constrained to derive its information via the net. It is also a less reliable approach to routing than to distribute the routing decisions among the IMPs.

The routing information cannot be propagated about the net in sufficient time to accurately characterize the instantaneous traffic flow. An efficient algorithm, therefore, should not focus on the movement of individual packets, but rather use topological or statistical information in the selection of routes. For example, by using an averaging procedure, the flow of traffic can be made to build up smoothly. This allows the routing algorithm ample time to adjust its tables in each IMP in advance of the build-up of traffic.

The scheme originally used in the ARPA network had each IMP select one output line per destination onto which to route packets. The line was chosen to be the one with minimum estimated time delay to the destination. The selection was updated every half second using minimum time estimates from the neighboring IMPs and internal estimates of the delay to each of the neighbors. Even though the routing algorithm only selects one line at a time per destination, two output lines will be used if a queue of packets waiting transmission on one line builds up before the routing update occurs and another line is chosen. Modifications to the scheme which allow several lines per destination to be used in an update interval (during which the routing is not changed) are possible using two or more time delay estimates to select the paths.

In practice, this approach has worked quite effectively with the moderate levels of traffic experienced in the net. For heavy traffic flow, this strategy may be inefficient, since the routing information is based on the length of queues, which we have seen can change much faster than the information about the change can be distributed. Fortunately, this information is still usable, although it can be substantially out of date and will not, in general, be helpful in making efficient routing decisions in the heavy traffic case.

A more intricate scheme, recently developed by BBN, allows multiple paths to be efficiently used even during heavy traffic.[18] Preliminary simulation studies indicate that it can be tailored to provide efficient routing in a network with a variety of heavy traffic conditions. This method separates the problem of defining routes onto which packets may be routed from the problem of selecting a route when a particular packet must be routed. By this technique, it is possible to send packets down a path with the fewest IMPs and excess capacity, or when that path is filled, the one with the next fewest IMPs and excess capacity, etc.

A similar approach to routing was independently derived by NAC using an idealized method that did not require the IMPs to participate in the routing decisions. Another approach using a flow deviation technique has recently been under study at UCLA.[13] The intricacies of the exact approach lead to a metering procedure that allows the overall network flow to be changed slowly for stability and to perturb existing flow patterns to obtain an increased flow. These approaches all possess, in common, essential ingredients of a desirable routing strategy.

Topological considerations

An efficient topological design provides a high throughput for a given cost. Although many measures of throughput are possible, a convenient one is the average amount of traffic that a single IMP can send into the network when all other IMPs are transmitting according to a specified traffic pattern. Often, it is assumed that all other IMPs are behaving identically and each IMP is sending equal amounts of traffic to each other IMP. The constraints on the topological design are the available common carrier circuits, the target cost or throughput, the desired reliability, and

TABLE I—23 Node 28 Link ARPA

Number of Circuits Failed	Number of Combinations to be Examined	Number of Cutsets
28	1	1
27	28	28
26	378	378
25	3276	3276
24	20475	20475
23	98280	98280
22	376740	376740
21	1184040	1184040
20	3108105	3108105
19	6906900	6906900
18	13123110	13123110
17	21474180	21474180
16	30421755	30421755
15	37442160	37442160
14	40116600	40116600
13	37442160	37442160
12	30421755	30421755
11	21474180	21474180
10	13123110	13123110
9	6906900	6906900
8	3108108	3108108
7	1184040	1184040
6	376740	349618
5	98280	≈70547
4	20475	≈9852
3	3276	827
2	378	30
1	28	0

(same)

the cost of computation required to perform the topological design.

Since, there was no clear specification of the amount of traffic that the network would have to accommodate initially, it was first constructed with enough excess capacity to accommodate any reasonable traffic requirements. Then as new IMPs were added to the system, the capacity was and is still being systematically reduced until the traffic level occupies a substantial fraction of the network's total capacity. At this point, the net's capacity will be increased to maintain the desired percentage of loading. At the initial stages of network design, the "two-connected" reliability constraint essentially determined a minimum value of maximum throughput. This constraint forces the average throughput to be in the range 10-15 kilobits per second per IMP, when 50 kilobit/sec circuits are used throughout the network, since two communication paths between every pair of IMPs are needed. Alternatively, if this level of throughput is required, then the reliability specification of "two-connectivity" can be obtained without additional cost.

Reliability computations

A simple and natural characterization of network reliability is the ability of the network to sustain communication between all operable pairs of IMPs. For design purposes, the requirement of two independent paths between nodes insures that at least two IMPs and/or circuits must fail before any pair of operable IMPs cannot communicate. This criterion is independent of the properties of the IMPs and circuits, does not take into account the "degree" of disruption that may occur and hence, does not reflect the actual availability of resources in the network. A more meaningful measure is the average fraction of IMP pairs that cannct communicate because of IMP and circuit failures. This calculation requires knowledge of the IMP and circuit failure rates, and could not be performed until enough operating data was gathered to make valid predictions.

To calculate network reliability, we must consider elementary network structures known as cutsets. A

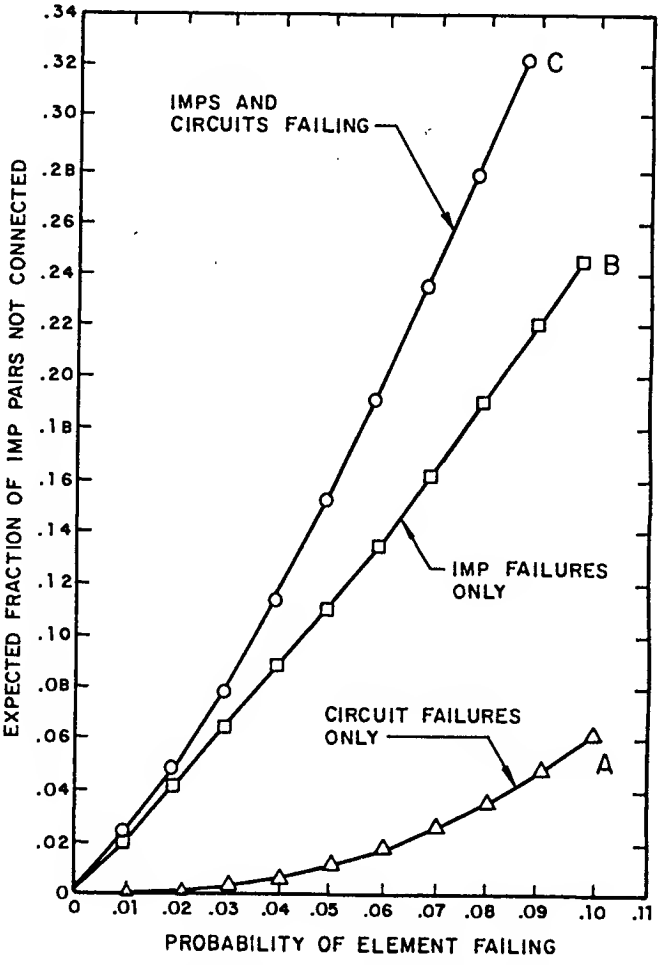

Figure 2—Network availability vs. IMP and circuit reliability

cutset is a set of circuits and/or IMPs whose removal from the network breaks all communication paths between at least two operable IMPs. To calculate reliability, it is often the case that all cutsets must be either enumerated or estimated. As an example, in a 23 IMP, 28 circuit ARPA Network design similar to the one shown in Figure 1(d), there are over twenty million ways of deleting only circuits so that the remaining network has at least one operable pair of IMPs with no intact communication paths. Table 1 indicates the numbers of cutsets in the 23 IMP network as a function of the number of circuits they contain.

A combination of analysis and simulation can be used to compute the average fraction of non-communicating IMP pairs. Detailed descriptions of the analysis methods are given in Reference 44 while their application to the analysis of the ARPANET is discussed in Reference 43. The results of an analysis of the 23 IMP version of the network are shown in Figure 2. The curve marked A shows the results under the assumption that IMPs do not fail, while the curve marked B shows the case where circuits do not fail. The curve marked C assumes that both IMPs and circuits fail with equal probability. In actual operation, the average failure probability of both IMPs and circuits is about 0.02. For this value, it can be seen that the effect of circuit failures is far less significant than the effect of IMP failures. If an IMP fails in a network with n IMPs, at least $n-1$ other IMPs cannot communicate with it. Thus, good network design cannot improve upon the effect directly due to IMP failures, which in the ARPANET is the major factor affecting the reliability of the communications. Further, more intricate reliability analyses which consider the loss of throughput capacity because of circuit failures have also been performed and these losses have been shown to be negligible.[28] Finally, unequal failure rates due to differences in line lengths have been shown to have only minor effects on the analysis and can usually be neglected.[27]

Topological optimization

During the computer optimization process, the reliability of the topology is assumed to be acceptable if the network is at least two-connected. The object of the optimization is to decrease the ratio of cost to throughput subject to an overall cost limitation. This technique employs a sophisticated network optimization program that utilizes circuit exchange heuristics, routing and flow analysis algorithms, to generate low cost designs. In addition, two time delay models were initially used to (1) calculate the throughput corresponding to an average time delay of 0.2 seconds, (2) estimate the packet rejection rate due to all buffers filling at an IMP. As experience with these models grew, the packet rejection rate was found to be negligible and the computation discontinued. The delay computation (Equation (7) in a later section) was subsequently first replaced by a heuristic calculation to speed the computation and later eliminated after it was found that time delays could be guaranteed to be acceptably low by preventing cutsets from being saturated. This "threshold" behavior is discussed further in the next section.

The basic method of optimization was described in Reference 12 while extensions to the design of large networks are discussed in Reference 9. The method operates by initially generating, either manually or by computer, a "starting network" that satisfies the overall network constraints but is not, in general, a low cost network. The computer then iteratively modifies the starting network in simple steps until a lower cost network is found that satisfies the constraints or the process is terminated. The process is repeated until no further improvements can be found. Using a different starting network can result in a different solution. However, by incorporating sensible heuristics and by using a variety of *carefully chosen* starting networks and some degree of man-machine interaction, "excellent" final networks usually result. Experience has shown that there are a wide variety of such networks with different topological structures but similar cost and performance.

The key to this design effort is the heuristic procedure by which the iterative network modifications are made. The method used in the ARPANET design involves the removal and addition of one or two circuits at a time. Many methods have been employed, at various times, to identify the appropriate circuits for potential addition or deletion. For example, to delete uneconomical circuits a straightforward procedure simply deletes single circuits in numerical order, reroutes traffic and reevaluates cost until a decrease in cost per megabit is found. At this point, the deletion is made permanent and the process begins again. A somewhat more sophisticated procedure deletes circuits in order of increasing utilization, while a more complex method attempts to evaluate the effect of the removal of any circuit before any deletion is attempted. The circuit with the greatest likelihood of an improvement is then considered for removal and so on.

There are a huge number of reasonable heuristics for circuit exchanges. After a great deal of experimentation with many of these, it appears that the choice of a particular heuristic is not critical. Instead, the speed and efficiency with which potential exchanges can be

investigated appears to be the limiting factor affecting the quality of the final design. Finally, as the size of the network increases, the greater the cost becomes to perform *any* circuit exchange optimization. Decomposition of the network design into regions becomes necessary and additional heuristics are needed to determine effective decompositions. It presently appears that these methods can be used to design relatively efficient networks with a few hundred IMPs while substantially new procedures will be necessary for networks of greater size.

The topological design requires a routing algorithm to evaluate the throughput capability of any given network. Its properties must reflect those of an implementable routing algorithm, for example, within the ARPANET. Although the routing problem can be formulated as a "multicommodity flow problem"[10] and solved by linear programming for networks with 20-30 IMPs,[8] faster techniques are needed when the routing algorithm is incorporated in a design procedure. The design procedure for the ARPA Network topology iteratively analyzes thousands of networks. To satisfy the requirements for speed, an algorithm which selects the least utilized path with the minimum number of IMPs was initially used.[12] This algorithm was later replaced by one which sends as much traffic as possible along such paths until one or more circuits approach a few percent of full utilization.[28] These highly utilized circuits are then no longer allowed to carry additional flow. Instead, new paths with excess capacity and possibly more intermediate nodes are found. The procedure continues until some cutset contains only nearly fully utilized circuits. At this point no additional flow can be sent. For design purposes, this algorithm is a highly satisfactory replacement for the more complicated multi-commodity approach. Using the algorithm, it has been shown that the throughput capabilities of the ARPA Network are substantially insensitive to the distribution of traffic and depend mainly only on the total traffic flow within the network.[8]

Analytic models of network performance

The effort to determine analytic models of system performance has proceeded in two phases: (1) the prediction of average time delay encountered by a message as it passes through the network, and (2) the use of these queueing models to calculate optimum channel capacity assignments for minimum possible delay. The model used as a standard for the average message delay was first described in Reference 21 where it served to predict delays in stochastic communication networks. In Reference 22, it was modified to describe the behavior of ARPA-like computer networks while in Reference 23 it was refined further to apply directly to the ARPANET.

The single server model

Queueing theory[20] provides an effective set of analytical tools for studying packet delay. Much of this theory considers systems in which messages place demands for transmission (service) upon a single communication channel (the single server). These systems are characterized by $A(\tau)$, the distribution of interarrival times between demands and $B(t)$, the distribution of service times. When the average demand for service is less than the capacity of the channel, the system is said to be stable.

When $A(\tau)$ is exponential (i.e., Poisson arrivals), and messages are transmitted on a first-come first-served basis, the average time T in the stable system is

$$T = \frac{\lambda \overline{t^2}}{2(1-\rho)} + \bar{t} \tag{1}$$

where λ is the average arrival rate of messages, $\bar{t}$ and $\overline{t^2}$ are the first and second moments of $B(t)$ respectively, and $\rho = \lambda \bar{t} < 1$. If the service time is also exponential,

$$T = \frac{\bar{t}}{1-\rho} \tag{2}$$

When $A(\tau)$ and $B(t)$ are arbitrary distributions, the situation becomes complex and only weak results are available. For example, no expression is available for T; however the following upper bound yields an excellent approximation[19] as $\rho \rightarrow 1$:

$$T \leq \frac{\lambda(\sigma_a^2 + \sigma_b^2)}{2(1-\rho)} + \bar{t} \tag{3}$$

where σ_a^2 and σ_b^2 are the variance of the interarrival time and service time distributions, respectively.

Networks of queues

Multiple channels in a network environment give rise to queueing problems that are far more difficult to solve than single server systems. For example, the variability in the choice of source and destination for a message is a network phenomenon which contributes to delay. A principal analytical difficulty results from the fact that flows throughout the network are correlated. The basic approach to solving these stochastic network